View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Purdue E-Pubs

Purdue University

Purdue e-Pubs

Department of Computer Science Technical Reports

Department of Computer Science

1993

Topologically Correct Approximations of Arbitrary Rational Parametric

Chandrajit L. Bajaj

Andrew V. Royappa

Report Number:
93-009

Bajaj, Chandrajit L. and Royappa, Andrew V., "Topologically Correct Approximations of Arbitrary Rational Parametric" (1993). *Department of Computer Science Technical Reports*. Paper 1028.
https://docs.lib.purdue.edu/cstech/1028

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries.
Please contact epubs@purdue.edu for additional information.

TOPOLOGICALLY CORRECT APPROXIMATIONS OF ARBITRARY RATIONAL PARAMETRIC

Chandrajit L. Bajaj
Andrew V. Royappa

CSD-TR-93-009
January 1993

Topologically Correct Approximations of Arbitrary Rational Parametric Surfaces

Chandrajit L. Bajaj
Andrew V. Royappa
Department of Computer Sciences

Purdue University
West Lafayette, IN 47907

December 31, 1992

Abstract

We study two problems related to approximating rational parametric surfaces. The first problem is how to construct a topologically correct polygonal mesh approximating the real part of an arbitrary rational parametric surface. The second problem is how to construct a topologically correct triangulation on an arbitrary rational parametric surface. In both problems, we place few restrictions on the rational functions defining the parametric surface. The rational functions are allowed to be undefined on an entire domain curve (the *pole curve*) and at certain special points (*base points*), and the surface is allowed to have nodal or cuspidal self-intersections. We also recognize that some real points on the parametric surface may be generated only by complex parameter values, and that some finite points on the surface may be generated only by infinite parameter values; we show how to compensate for these conditions. We give an algorithm for handling these problems which has applications in mathematical visualization, rendering arbitrary NURBS, and in finite-element meshing. The main idea of the algorithm is to partition the parametric domain into disjoint regions along certain curves. We construct a domain triangulation that "respects" these regions and then map it onto the surface; the resulting mesh of triangles on the surface satisfies the topological correctness and triangulation properties.

1 Introduction

In this paper we give an algorithm for computing a topologically correct polygonal approximation to the real part of any rational parametric surface.

Points on a parametric surface patch can be generated by sampling the parametric functions over some region of the parameter domain. Because of this, the display of patches of parametric surfaces is well-understood [16, 42, 33, 25, 41, 45]. Some methods address in detail the problem of generating a polygonal mesh on a surface that is sensitive to variations in surface curvature: view-dependent methods [49] as well as view-independent [32, 38, 8].

The parametric functions that define a surface can be viewed as a map from $\mathcal{R}^2$ into $\mathcal{R}^3$. "Domain sampling" methods such as the above assume that the parametric functions are defined and continuous in the region of the parameter domain that is being mapped. If the parametric functions are *rational*, however, they could be undefined at some points in $\mathcal{R}^2$. Many surfaces (including simple ones such as some quadrics) are given by rational maps which are undefined at some points.

We investigate how to correctly approximate a part of an *arbitrary* parametric surface, given a rational map that defines the surface. Our techniques are applicable whether this part of the surface is described by a bounded portion of the parameter domain, or by a bounding box in $\mathcal{R}^3$. If a bounding box is specified the algorithm will use the entire (infinite) parameter domain to compute parts of the surface that lie inside the box.

In this formulation the problem is of interest to CAD designers as well as mathematicians interested in surface visualization. The former usually express the rational functions defining the surface in terms of the rational Bezier or B-spline bases [14] with non-negative weights, restricting the rational functions to a standard part of the domain.

However, researchers are considering generalizations to rational patches in which the rational functions are not defined everywhere [22, 50], making our techniques relevant.

In addition to topologically correct approximations, we consider the problem of constructing *triangulations* on arbitrary rational parametric surfaces, especially surfaces that self-intersect. Constructing triangulations on surfaces is useful for mesh generation in finite-element analysis. It turns out that our surface approximation technique can be extended in a straightforward way to handle this useful companion problem.

Thus our surface approximation techniques find application in the mathematical visualization of surfaces (our original motivation), rendering of arbitrary NURBS, and in finite-element meshing.

Let a parametric surface be given by a rational map of three rational functions:

$$x(s,t) = \frac{X(s,t)}{W(s,t)}, \quad y(s,t) = \frac{Y(s,t)}{W(s,t)}, \quad z(s,t) = \frac{Z(s,t)}{W(s,t)} \tag{1}$$

where X, Y, Z, W are polynomials with real coefficients and no common factor. Then we formulate two problems as follows:

- Given a portion of the domain, compute a topologically correct piecewise-linear approximation to the corresponding part of the surface defined by (1). Or, given a bounding box in $\mathcal{R}^3$, compute a topologically correct piecewise-linear approximation to the parts of the surface lying inside the box. In this paper we assume the latter case of the problem since the techniques to be discussed apply to the former case also.
- As above, except we further require the piecewise-linear approximation to be a surface *triangulation*, i.e. it must be a triangular mesh whose edges meet only at vertices and along edges.

Our basic approach to constructing a surface triangulation is to map a domain triangulation onto the surface using the parametric map, as in common. However, the domain triangulation is constructed carefully so that the surface triangulation is topologically correct.

In this paper we describe various subproblems that arise in trying to solve the above problems when we don't place any restrictions on the rational functions $x(s,t)$,$y(s,t)$,$z(s,t)$. We then give a solution for each subproblem, and combine the solutions in an algorithm for generating topologically correct triangulations on arbitrary rational parametric surfaces.

The subproblems are explained in detail in section 3. They are: domain poles, domain base points, surface self-intersections, complex parameter values, and infinite parameter values. We describe them briefly here.

1. **Domain poles.** The map is undefined at points satisfying $W(s,t) = 0$. There is a one-dimensional family of such domain points. The parametric functions can't be evaluated at such points; even if they never are, we might construct a surface approximation that does not represent its shape correctly.

2. **Domain base points.** The map is undefined at points satisfying $X(s,t) = Y(s,t) = Z(s,t) = W(s,t) = 0$. There are finitely many such points, called domain *base points.* It is known that an entire curve on the parametric surface corresponds to each base point; the points of this curve can't be directly computed using the rational map. Ignoring base points can lead to a topologically incorrect surface approximation.

3. **Surface self-intersections.** The surface intersects itself. Even if the rational map has no poles or base points, mapping an arbitrary domain triangulation onto a parametric surface may not yield a surface triangulation because surface triangles cross each other.

4. **Complex parameter values.** Some real points of the surface are generated only by complex parameter values.

5. **Infinite parameter values.** Some finite points of the surface are generated only by infinite parameter values.

For graphics display and NURBS rendering, subproblem (3) is not necessary (although z-buffering still causes wavy lines along polygon intersections due to aliasing). If finite-element meshing is the application, subproblem (3) is of interest.

The problems can be extended to include rational parametric surfaces in higher dimensions, but we don't discuss this here. The general flavor of the methods discussed will still apply, although implementing higher-dimensional methods would require more tools.

In a preliminary paper [9] we discussed subproblems (1) and (5). In the current paper we give solutions for (2), (3), and (4) as well. Because of this, the current paper has a much broader scope and more applications than [9].

The rest of this paper is organized as follows. First, we discuss two approaches: either directly approximating the surface in the range space of the parametric functions, or computing those portions of the domain that map onto the desired parts of the surface. We argue that the domain-space approach is preferable in this context. After explaining the above subproblems in detail, we present techniques for dealing with each subproblem. We then use these techniques in an algorithm for generating topologically accurate surface triangulations. After explaining the algorithm detail we discuss situations in which it can fail and where it could be improved, based on extensive experimentation.

2 Domain space vs. range space approaches

One way to construct a piecewise-linear approximation to a parametric surface is to evaluate the parametric functions at various points on the parameter domain, and link together the resulting surface points to form an approximating mesh. When considering arbitrary rational parametric surfaces, the parametric functions may not be defined at some points, since rational functions are not defined at points where the denominator vanishes. Such points are called *poles*, and usually correspond to surface points at infinity. The exception occurs when all the polynomials X, Y, Z, W vanish there (an event that can happen only finitely many times since they have no common divisor, by assumption). In this case the parameter point is a domain *base point*.

We shall later explore poles and base points in detail, showing examples of how they can cause domain sampling techniques to fail.

Another way to approach the problem is to work directly in the range space of the rational function map. Since we are only interested in portions of a surface inside a bounding box, and poles correspond to surface points at infinity, a range-space method can avoid explicitly evaluating the rational functions at poles (base points still cause problems).

The following system of equations is equivalent to (1):

$$\begin{aligned} W(s,t)x - X(s,t) &= 0 \\ W(s,t)y - Y(s,t) &= 0 \\ W(s,t)z - Z(s,t) &= 0 \end{aligned}$$

One can theoretically implicitize the parametric surface by eliminating s, t from this system [34] using several available methods [15, 19, 18, 26, 37] and then approximate the resulting implicit surface directly. Note that a parametric surface of degree n could have an implicit equation of degree n^2.

However, implicit surface approximation techniques [7, 12, 13, 19, 28, 39, 48] don't handle surface self-intersections very well, although research is being done to overcome this [1, 2, 11]. Since we would like to display surfaces with complicated singularities and several real sheets, we avoid the range-space approach. We show instead that a careful evaluation of the domain is sufficient to generate an accurate piecewise-linear approximation of the parametric surface.

3 Difficulties in domain sampling

In this section we explain why domain base points and poles sometimes cause sampling techniques to fail, and give simple examples that are representative of the kinds of failures that occur. The main problem is that domain sampling techniques which don't take poles and base points into account can generate surface approximations which do not accurately represent the topology of the surface.

3.1 Domain poles

Inability to evaluate a rational function at a pole (i.e., generating a divide by zero exception in a numerical program) is not the main reason that domain sampling methods fail when poles are present. Even if a domain sampling method avoids evaluating a rational map at a pole, it may construct an approximation that does not reflect the actual shape of the surface. This happens when a part of the domain that contains a pole is mapped onto the surface.

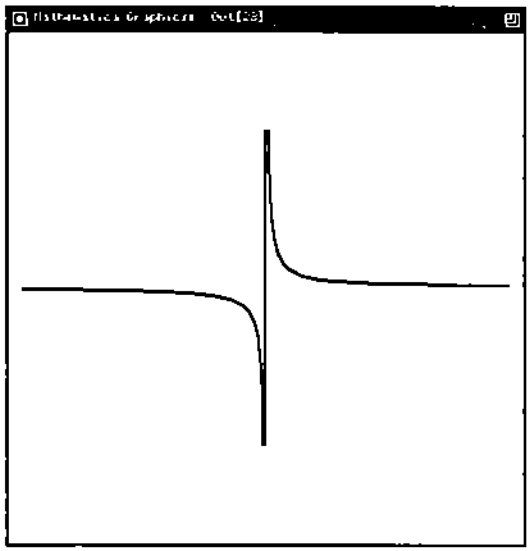

Figure 1: Disjoint branches being wrongly connected (*Mathematica*)

When a parameterization contains poles, the surface may have multiple branches or sheets. We show a simple example using a parametric curve. The hyperbola given by

$$x(s) = s, \quad y(s) = \frac{1}{s}$$

has a pole at $s = 0$. The real part of the curve consists of two branches.

A simple domain sampling algorithm for approximating this hyperbola might select a closed interval $[a, b]$ in the parameter domain, generate n equally-spaced parameter values $s_i = a + i(\frac{b-a}{n-1}), i = 0, \ldots, n$, and then connect the points $(x(s_{i-1}), y(s_{i-1}))$, $(x(s_i), y(s_i))$ with a straight-line segment. In this example, a line segment could be drawn between points whose parameter values lie on opposite sides of a pole. As a result, the approximation does not accurately represent the shape of the curve. Figure 1 shows the output of the program *Mathematica* for plotting the hyperbola over the domain interval $s \in [-\frac{1}{2}, \frac{1}{2}]$.

With surfaces the problem is acute, and poles can cause problems even when the surface has a single real sheet. For instance, a hyperboloid of one sheet with implicit equation $x^2 + y^2 - z^2 - 1 = 0$ is a surface whose real part is single-sheeted (i.e. connected). However, if we work from the equivalent parametric representation

$$x(s,t) = \frac{t^2 - s^2 + 1}{s^2 + t^2 - 1}, \quad y(s,t) = \frac{2st}{s^2 + t^2 - 1}, \quad z(s,t) = \frac{2t}{s^2 + t^2 - 1} \tag{2}$$

then problems arise because of the pole curve described by $s^2 + t^2 - 1 = 0$ in the parameter domain. The right picture in Figure 2 shows the output produced by *MapleV* for this surface with $(s,t) \in [-2,2] \times [-2,2]$ (a domain region containing the pole curve).

A small digression is in order about the programs *Mathematica* and *MapleV*. Both programs use sophisticated strategies for graphing curves and surfaces that are generally effective. However, they use domain sampling techniques which are not equipped to handle parametric functions that are not defined everywhere, and hence fail for simple examples such as the above.

3.2 Domain base points

We assumed that the numerators and common denominator of the rational map (1) have no common factor. It is still possible that there are a finite number of points (a, b) such that $X(a, b) = Y(a, b) = Z(a, b) = W(a, b) = 0$. Each such point is called a *base point* of the parametric surface. Information about base points can be found in books

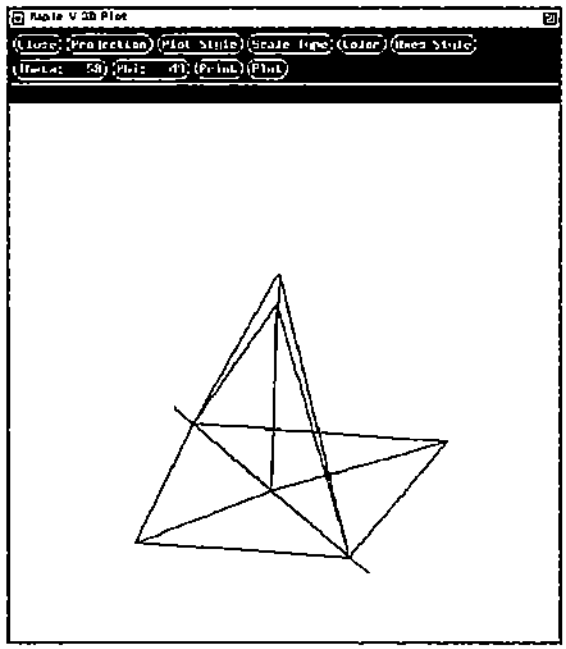

Figure 2: Single-sheeted surface with domain poles (*Maple*)

on algebraic geometry such as [29, 44, 53]. Interesting material on base points in the context of CAGD appears in [18, 36, 43, 50]. In particular, [50] shows how to represent patches with up to six sides in the triangular rational Bezier patch form, by a clever use of domain base points.

Base points are problematic since there is no one surface point for the corresponding domain point. To each base point there actually corresponds a rational curve on the surface [44]. Approaching the base point along different directions leads to different points on the surface; the points corresponding to all directions form a space curve that lies on the surface. Since there is no parameter value for points on this curve (at which the surface map is defined), the entire curve will be missing from the parametric surface. Such a curve is called a *seam curve*. Even if poles are taken care of in some way, the seam curves can show up as *gaps* on the surface, as in Figure 3. This figure shows the hyperboloid of one sheet given by (2). This parameterization has the two base points $(s,t) = (\pm 1, 0)$. The corresponding seam curves can be parameterized in parameters u, v, giving the lines $(x(u), y(u), z(u)) = (-1, u, u)$ and $(x(v), y(v), z(v)) = (-1, v, -v)$ on the surface.

If base points are not taken into account, the domain sampling density may need to be unnecessarily dense (with respect to surface curvature) in order for the gaps to be narrow. Furthermore, even if the gap is narrow enough to suffice for display, the surface approximation will not correctly represent the surface's topology because of the gap.

3.3 Surface self-intersections

A triangular mesh on a parametric surface is derived by constructing a planar triangulation in the domain and mapping it onto the surface. However, when a planar domain triangulation is mapped onto a curved surface, the resulting triangles in space may no longer form a triangulation.

There are two reasons for this. First, if the domain sampling density is not fine enough with respect to the surface curvature, two surface triangles may overlap each other. Second, if the surface actually crosses itself, some surface triangles near the crossing may cross each other. For finite-element mesh generation, surface triangulations are preferred. Even for display, a surface triangulation is preferable. This is because scanline-rendering algorithms suffer from aliasing effects along triangle intersections; this causes what should appear as a sharp edge on the screen to appear wavy.

Figure 4 shows a triangular mesh approximating the a Steiner quartic surface (Figure 7). The mesh was con-

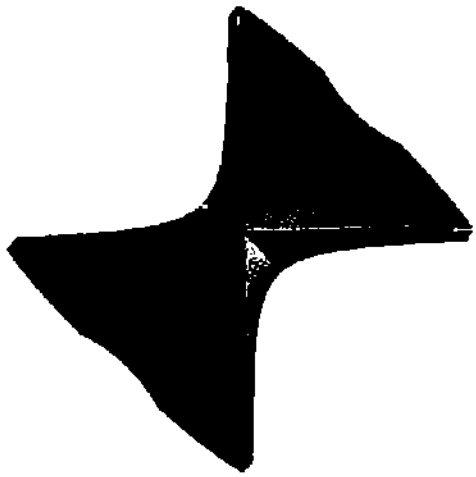

Figure 3: Hyperboloid of 1 sheet with seam curve gaps

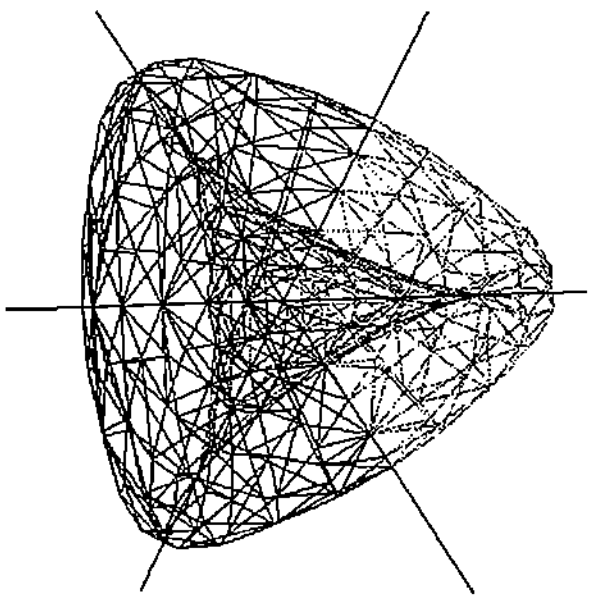

Figure 4: Triangulation of a Steiner surface along line singularities

structed using the surface display algorithm. The surface crosses itself along the x,y and z axes. In this case, the mesh is actually a surface triangulation. This is a coincidence, and happens because the four quadrants of the parameter domain happen to map onto four pieces of the surface that meet exactly along the singular lines. Since the surface display algorithm maps each of the four domain quadrants separately, the resulting triangles on the mesh also meet along the singular lines.

Suppose we apply a random linear reparameterization to get another map for the same Steiner surface, and apply the display algorithm to generate a mesh for the new map. In general, the new mesh will not be a surface triangulation.

3.4 Complex parameter Values

While the parameterization (1) defines a map from $\mathcal{R}^2$ into $\mathcal{R}^3$, it also defines a unique algebraic surface in $\mathcal{C}^3$ which can be given by a single equation in three variables, with real coefficients. This algebraic surface may contain real points which are not mapped by any real parameter values. If we want to view the entire real part of the algebraic surface defined by the map, and not just the image of $\mathcal{R}^2$, additional computations are needed.

For instance, consider a Steiner surface (Figure 7) given implicitly by $F(x,y,z) = x^2y^2 + y^2z^2 + x^2z^2 - 2xyz = 0$, or parametrically by

$$x(s,t) = \frac{2s}{s^2+t^2+1}, \quad y(s,t) = \frac{2t}{s^2+t^2+1}, \quad z(s,t) = \frac{2st}{s^2+t^2+1}$$

(it is a quartic algebraic surface defined by a quadratic rational map).

Note that the x, y and z axes lie entirely on the algebraic surface $F(x,y,z) = 0$. Let us consider the parametric map to see which parameter values give rise to the x axis, which is described by $y = z = 0$. Setting $y(s,t) = z(s,t) = 0$ and solving for s,t yields $t = 0$. Thus $(x(s,0),0,0) = (2s/(s^2+1),0,0)$, $s \in \mathcal{R}$, are the points on the x axis that are given by the map. This shows that any parameter value $s \in \mathcal{R}$ yields a surface point $(x,0,0)$ with $|x| \leq 1$.

To find parameter values giving rise to the remaining surface points on the x-axis we must extend the parameter domain to $\mathcal{C}^2$.

3.5 Infinite parameter values

Consider the following map for the unit sphere in $\mathcal{R}^3$:

$$\begin{aligned} x &= \frac{1-s^2-t^2}{s^2+t^2+1} \\ y &= \frac{2s}{s^2+t^2+1} \\ z &= \frac{2t}{s^2+t^2+1} \end{aligned}$$

The (finite) point $(-1,0,0)$ on the sphere is the image of the entire line at infinity in $\mathcal{R}^2$. Simply using large parameter values to represent infinity is not enough to construct a topologically correct polygonal approximation; as in Figure 5, the polygons will approach the "missing point" ever closer but never fill the gap.

To compute certain finite points on the surface we may need to extend the parameter domain to include parameter values at infinity, i.e. extend the parameter domain to be the projective plane.

4 Summary

We have described the main problems that occur in constructing topologically accurate polygonal meshes on rational parametric surfaces, when no restriction is placed on the rational map defining the surface.

These problems generally occur because a particular rational map for the surface can be locally "bad" near some domain points. However, from any single map for the parametric surface we can extract all information necessary to compute all parts of the surface inside the bounding box.

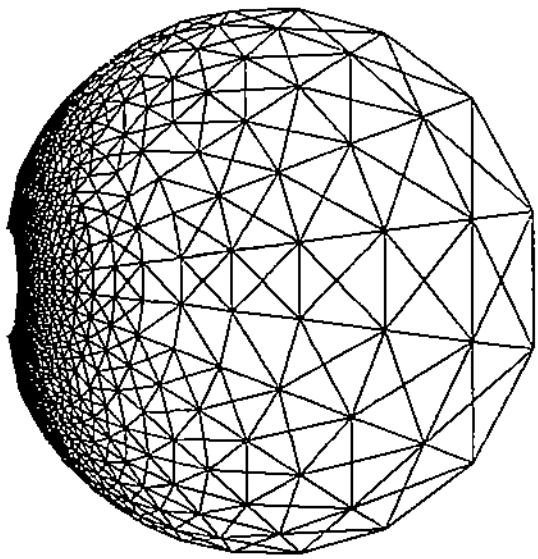

Figure 5: Infinite parameter values mapping to finite point

5 Techniques for overcoming difficulties

In this section we outline the basic idea for solving each of the problems addressed above. Additional details are given in the next section, when the complete algorithm is shown.

5.1 Partition of domain by pole curves

Rational functions are undefined at points in the domain where their denominator vanishes, and continuous everywhere else. Hence, the pole curve partitions the parameter domain into regions, such that inside each (open) region the functions of the parametric map are defined and continuous.

Therefore, our approach to handling pole curves is simple: we partition the domain by the pole curve. In particular, we construct a special triangulation of the domain that respects this partition. In this triangulation, a domain triangle contains pole points only on its boundary and not in its interior. Since pole curves may not be lines, in practice we shall construct a piecewise-linear approximation of the pole curve and then identify linear curve approximants with edges of the triangulation.

Once such a triangulation is constructed, we know that each domain triangle maps onto a single-sheeted patch, since there are no pole points in the interior (pole points at a vertex correspond to points at infinity, and therefore the patch may be semi-infinite). A conventional domain sampling technique is used in the interior of the triangle to mesh the patch to any desired precision. The patch can then be clipped against a bounding box, if necessary.

If base points are not present, domain partitioning combined with the handling of infinite parameter values (discussed below) suffices to generate a topologically correct mesh of the parametric surface, even if it is multi-sheeted.

5.2 Base points and seam curve parameterizations

When base points are present, it is not sufficient to just handle pole curves as gaps may still be present, as in Figure 3. The surface approximation will then not be topologically correct, since the surface approximation will be "torn" along the seam curves.

To handle base points, we must "stitch" the surface up along seam curves. This can be done in the framework of domain partitioning, as follows. We compute all base points and insert them into the domain triangulation as

additional vertices – thus base points will occur explicitly at the vertices of a domain triangle.

In general, approaching a base point along different directions in the domain leads to a different surface point (in the limit). Thus a base point "blows up" onto an entire "seam" curve on the surface [44] – each point of this curve corresponds to a different limit direction at the base point. A consequence of this fact is that a domain triangle with a base point vertex maps onto a four-sided patch on the surface. In general, a triangle with b base point vertices maps onto a $(b+3)$-sided patch – a fact exploited in [50] to represent multi-sided patches over triangular domains.

Once we have a *parameterization* of the seam curves, it is easy to generate the patch corresponding to a domain triangle with base point vertices, however many sides it has. Each of the two edges adjacent to a base point vertex corresponds to a particular direction, and therefore to a particular parameter value. The two parameter values then define a segment of the seam curve. This curve segment is the side on the patch that corresponds to the domain base point.

We now discuss the computation of seam curve parameterizations. Points on a rational parametric surface are given as follows (temporarily using projective coordinates for notational convenience):

$$\rho X = X(s,t), \quad \rho Y = Y(s,t), \quad \rho Z = Z(s,t), \quad \rho W = W(s,t)$$

where ρ is a non-zero constant of proportionality (we still use an affine domain, which is sufficient as we later show).

Then, let O be a common solution of the curves $X = 0, \ldots, W = 0$. Furthermore, let us suppose that O is a point of multiplicity q on each of the curves $X = 0, \ldots, W = 0$, and that the curves have no common tangent at O. Then the image of the base point O is a rational curve of degree q on the surface [44].

In [18], a method is given to find the parametric equations of this curve. The basic idea is to pass a pencil of lines through the base point and then use the slope of these lines as a parameter, since approaching the base point from each direction leads to a different point on the seam curve. The seam curve equations are not given explicitly, but as quotients of certain polynomials. The algorithm fails when the curves $X = 0, \ldots, W = 0$ have common tangents at O; in this case the parametric equations given by this algorithm generate only a single point of the seam curve.

In [36] a method is given for parameterizing seam curves that works for all cases (i.e., even when the tangents are equal). However, it is much more expensive than the previous method and not currently practical: multivariate resultants are used to compute a projection onto a plane of all the seam curves simultaneously, yielding a bivariate equation. Along with the projection, a rational map R is computed between the projection and the curves on the surface. A bivariate factorization algorithm (over the complexes) such as [4, 30] must first be applied to separate out the the projections of the individual curves. Each projected seam curve is then parameterized using a general curve parameterization technique [3], and finally mapped onto the surface using the rational map M.

The method of [18] is much simpler than that of [36], and could be implemented as part of the surface display algorithm. However, we present a further simplification of [18] based on the the same idea, which is found in algebraic geometry textbooks such as [44] (and hence it also fails when the tangents at O are all equal). This simplification makes the method easier to implement numerically, since we find an explicit formula for the parametric equations of the seam curve. Furthermore, the formula clearly shows how the number of common tangents of $X = 0, \ldots, W = 0$ at the base point affects the seam curve, explaining why this method breaks down when the tangents at the base point are all equal.

THEOREM 1 *Let (a,b) be a base point of multiplicity q. Then for any $m \in \mathcal{R}$, the image of a domain point approaching (a,b) along a line of slope m is given by* $(X(m), Y(m), Z(m), W(m)) =$

$$\left(\sum_{i=0}^{q} \left(\frac{\partial^q X}{\partial s^{q-i} \partial t^i}(a,b) \right) \binom{q}{i} m^i, \ldots, \sum_{i=0}^{q} \left(\frac{\partial^q W}{\partial s^{q-i} \partial t^i}(a,b) \right) \binom{q}{i} m^i \right) \tag{3}$$

PROOF. Consider the image of a point (s,t) as it approaches (a,b) along the line of slope m through (a,b). Expressing the line as $t = m(s-a)+b$, this yields the point

$$\lim_{s \to a} (X(s, m(s-a)+b), Y(s, m(s-a)+b), Z(s, m(s-a)+b), W(s, m(s-a)+b)) \tag{4}$$

Expanding $X(s,t)$ in a Taylor series at (a,b) yields

$$X(s,t) = \sum_{k=0}^{p} \sum_{i=0}^{k} \frac{(s-a)^i (t-b)^{k-i} \binom{k}{i}}{k!} \frac{\partial^k X}{\partial s^i \partial t^{k-i}}(a,b) \tag{5}$$

Substituting $t = m(s-a)+b$ in (5) yields

$$\begin{aligned} X(s) &= \sum_{k=0}^{p}\sum_{i=0}^{k} \frac{(s-a)^k \binom{k}{i} m^{k-i}}{k!} \frac{\partial^k X}{\partial s^i \partial t^{k-i}}(a,b) \\ &= (s-a)^q \sum_{k=q}^{p}\sum_{i=0}^{k} \frac{(s-a)^{k-q} \binom{k}{i} m^{k-i}}{k!} \frac{\partial^k X}{\partial s^i \partial t^{k-i}}(a,b) \end{aligned}$$

where q is the multiplicity of the base point (a,b), which implies that all derivatives of $X(s,t)$ up to order $q-1$ vanish at (a,b).

Substituting $t = m(s-a)+b$ into the Taylor expansions of $Y(s,t), Z(s,t), W(s,t)$ yields $(X(s),\ldots,W(s)) =$

$$(s-a)^q \left(\sum_{k=q}^{p}\sum_{i=0}^{k} \frac{(s-a)^{k-q} \binom{k}{i} m^{k-i}}{k!} \frac{\partial^k X}{\partial s^i \partial t^{k-i}}(a,b), \ldots, \sum_{k=q}^{p}\sum_{i=0}^{k} \frac{(s-a)^{k-q} \binom{k}{i} m^{k-i}}{k!} \frac{\partial^k W}{\partial s^i \partial t^{k-i}}(a,b) \right)$$

We drop the factor of proportionality $(s-a)^q$ and compute the limit (4):

$$\begin{aligned} \lim_{s\to a}(X(s),Y(s),Z(s),W(s)) &= \frac{1}{q!}\left(\sum_{i=0}^{q} \binom{q}{i} m^{q-i} \frac{\partial^q X}{\partial s^i \partial t^{q-i}}(a,b), \ldots, \sum_{i=0}^{q} \binom{q}{i} m^{q-i} \frac{\partial^q W}{\partial s^i \partial t^{q-i}}(a,b) \right) \\ &= \left(\sum_{i=0}^{q} \left(\frac{\partial^q X}{\partial s^{q-i}\partial t^i}(a,b) \right) \binom{q}{i} m^i, \ldots, \sum_{i=0}^{q} \left(\frac{\partial^q W}{\partial s^{q-i}\partial t^i}(a,b) \right) \binom{q}{i} m^i \right) \end{aligned}$$

Thus for each $m \in R$ there is a corresponding point (3) on the parametric surface. These points collectively form a one-dimensional family or curve on the surface. □

COROLLARY 1 *If the curves $X(s,t) = 0, \ldots, W(s,t) = 0$ share t tangent lines at (a,b), then the seam curve $(X(m), Y(m), Z(m), W(m))$ has degree $q-t$. In particular, if $X(s,t) = 0, \ldots, W(s,t) = 0$ have identical tangents at (a,b), then for all $m \in R$ the coordinates $(X(m),\ldots,W(m))$ represent a single point.*

PROOF. The equations of the tangent lines to the curve $X(s,t) = 0$ at (a,b) are given by equating to zero the factors of the following curve, which are all linear (since it is homogeneous):

$$\sum_{i=0}^{q} \left(\frac{\partial^q X}{\partial s^{q-i}\partial t^i}(a,b) \right) \binom{q}{i} s^{q-i} t^i = 0 \tag{6}$$

and similarly for $Y(s,t) = 0$ etc. Moreover, there is a 1-1 correspondence between the linear factors of this curve and the roots of the polynomial $X(m)$ in (3). Thus each common tangent of $X(s,t) = 0, \ldots, W(s,t) = 0$ at (a,b) leads to a common root, and hence a common factor, among $X(m),\ldots,W(m)$. If there are t common tangents there will be a common factor of degree t, which can be divided out of the seam curve parameterization $(X(m),\ldots,W(m))$ since proportional homogeneous coordinates represent the same point. Thus the seam curve is of degree $q-t$. □

5.3 Partitioning along surface self-intersections

Earlier, we mentioned two reasons why a domain triangulation might not stay a triangulation when it is mapped onto a parametric surface. The first reason was because the domain sampling density was not high enough, and the second reason was because the surface might self-intersect.

The first case can be handled by increasing the domain sampling density (either locally or globally, although local curvature-sensitive sampling is much preferred since it generates fewer polygons). Several domain sampling techniques already adjust the sampling density due to curvature, so we focus on the second case.

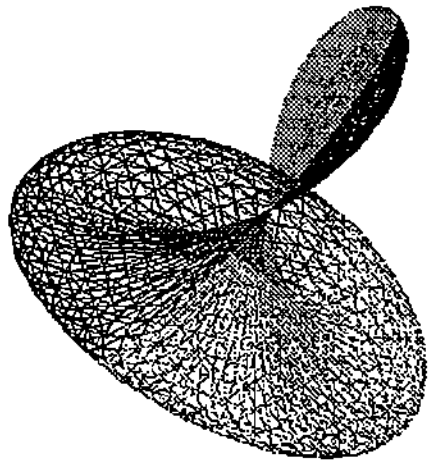

Figure 6: Triangulation of parametric surface with point singularity

The domain-partitioning technique lends itself to generating triangulations on surfaces that self-intersect. The key idea is to compute those points and curves in the parametric domain that map onto surface self-intersections, and then partion the domain by these points and curves (as well as by the pole curves). If this is done, no domain triangle will contain in its interior a point that map onto a surface singularity. Hence, triangles on the surface will meet only along their edges or at their vertices, even if the surface is singular.

Domain curves (and points) mapping onto surface singularities can be computed by solving systems of polynomial equations. For instance, cuspidal singularities correspond to domain points where the Jacobian matrix of the rational map does not have full rank. We can compute the symbolic Jacobian matrix and equate its minors to zero, yielding a set of polynomial equations whose common solution are domain points that map onto surface cusps. Nodal singularities can also be computed by solving a system of polynomial equations.

The system of equations has a one-dimensional solution set in general. Multivariate resultants [34, 35, 5, 37] can be used to project the solutions onto the parameter plane, after which a curve-tracer can be used to compute an approximation. For tracing the curve one can use either subdivision methods, e.g. [27], or a marching method such as [6].

For example, consider the surface given by the the following equations, taking $x(s,t) = X(s,t)/W(s,t)$, etc.

$$\begin{aligned} X(s,t) &= s^3 + st^2 - 3s \\ Y(s,t) &= (s^2+t^2)^2 - 3(s^2+t^2) \\ Z(s,t) &= s^2t + t^3 - 3t \\ W(s,t) &= (s^2+t^2)^2 + 2(s^2+t^2) + 1 \end{aligned}$$

By substitution, one can verify that its implicit equation is

$$F(x,y,z) = z^4 + 2y^2z^2 + 3yz^2 + 2x^2z^2 + y^4 - y^3 + 2x^2y^2 + 3x^2y + x^4 = 0$$

This is a surface of revolution (see Figure 6); it has a point singularity at the origin.

It can be shown that the domain points mapping onto the surface singularity satisfy $(t^2 + s^2 - 3)(t^2 + s^2) = 0$. Thus the circle of radius $\sqrt{3}$ centered at the origin, and the origin itself both map onto the surface (nodal) self-intersection at $(0,0,0)$. This circle and the origin partition the parameter domain into regions that meet at the

surface self-intersection. By partitioning the parameter domain by the curve $t^2 + s^2 - 3 = 0$ and the point $(0, 0)$, as by pole curves, we can construct a triangulation on this surface.

5.4 Computing complex parameter values

We now show one way to compute the complex parameter values that map onto these points. Let the parameters s, t denote complex numbers given as $s = a + bi$, $t = c + di$, where $a, b, c, d \in \mathcal{R}$ and $i = \sqrt{-1}$.

Then the parametric map from $\mathcal{C}^2 \rightarrow \mathcal{R}^3$ can be expressed as

$$\begin{aligned} x(s,t) = x(a+bi, c+di) &= X_R(a,b,c,d) + X_I(a,b,c,d) \cdot i \\ y(s,t) = y(a+bi, c+di) &= Y_R(a,b,c,d) + Y_I(a,b,c,d) \cdot i \\ z(s,t) = z(a+bi, c+di) &= Z_R(a,b,c,d) + Z_I(a,b,c,d) \cdot i \end{aligned}$$

where X_R denotes the real part of $x(a + bi, c + di)$ and X_I denotes its imaginary part, etc.

Then $X_I(a, b, c, d) = 0$, $Y_I(a, b, c, d) = 0$, $Z_I(a, b, c, d) = 0$ form a system of three equations in four unknowns whose solutions give parameter values that map to real surface points. In general, such a system has a one-dimensional solution set.

Note that this particular system has the trivial two-dimensional solution $b = d = 0$ which must be excluded. Thus the marching method [6] cannot be used directly; rather, as for surface self-intersections, we must use resultants to first compute a projection of the space curve. After deleting the extraneous component due to the trivial solution, we can trace the projected plane curve and finally map it onto the space curve using the inverse of the projection.

The points (a, b, c, d) of the space curve give complex parameter values $s = a + bi$ and $t = c + di$ that map onto real points of the surface.

5.5 Mapping infinity using projective reparameterization

To handle infinite parameter values, we use projective reparameterizations. In [20], a technique called "homogeneous sampling" is used to sample finite and infinite points of a surface equally. We use similar idea based on projective reparameterizations, so that only affine parameter values are needed. Specializing theorem 1 of [10], we use four reparameterizations of the original rational map, given by

$$\begin{aligned} s &= \pm\frac{u}{1-u-v} \\ t &= \pm\frac{v}{1-u-v} \end{aligned}$$

Each reparameterized map needs to be sampled only over the unit triangle of its domain ($u \geq 0$, $v \geq 0$, $u+v \leq 1$), yielding a triangular patch. The patches meet along their boundaries and together cover the entire surface (including finite points that were generated by infinite parameter values in the original surface).

Figure 7 shows a member of the Steiner surface family mapped using four reparameterizations. Each piece is the image of a different domain quadrant under the original rational map. Each piece is given a different color.

6 The algorithm

As is common in domain sampling techniques, a triangulation of the parametric domain is mapped onto the surface, yielding a piecewise-linear approximation to it. Triangular surface elements have several advantages, described in detail in [49].

The pole curve partitions the parameter domain into several regions. The rational functions of the map are continuous inside these open regions, and therefore each region maps to a possibly infinite but *single sheeted* surface patch. By approximating each patch independently of the others, we avoid generating a topologically incorrect approximation.

The algorithm uses the pole curve and any self-intersection image curves to partition the domain into regions. This is done by generating a piecewise-linear approximation to the pole curve and the self-intersection image curves. A triangulation is then constructed of the curve points, base points and sufficiently many other ordinary domain

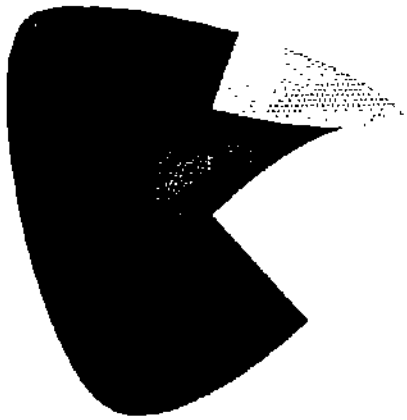

Figure 7: Total mapping of rational parametric surface (steiner surface)

points. Some processing is done to ensure that none of the triangle edges crosses the pole curve or a self-intersection image curve.

The domain triangulation "respects" the pole curves and self-intersection image curve. In other words, the interior of a domain triangle in this triangulation maps onto a patch of the surface that is single-sheeted and does not intersect itself. Figure 8 shows a domain triangulation respecting a pole curve.

Each domain triangle is allowed to have up to two vertices that are on the pole curve (note that base points are also on the pole curve). Each domain triangle then corresponds to a single-sheeted surface patch.

We first show the steps of the algorithm, and then explain the steps in detail.

1. (RESTRICT TO FINITE DOMAIN) Perform a projective reparameterization so that the entire surface is mapped in four pieces, each over the "unit" triangle spanned by $(0,0),(0,1),(1,0)$. Treat the four new mappings independently, and for each mapping perform the following steps.

2. (GENERATE POLE POINTS) Compute a piecewise-linear approximation to the pole curve of the current mapping inside the unit triangle.

3. (GENERATE SELF-INTERSECTION IMAGE POINTS) Compute a piecewise-linear approximation to the image curves or points of any surface self-intersection.

4. (GENERATE BASE POINTS) Compute all the base points of the current mapping that lie inside the unit triangle.

5. (GENERATE DOMAIN POINTS) Generate points in the rest of the unit triangle according to some fixed or adaptive scheme.

   We label each kind of point accordingly.

6. (TRIANGULATE) Compute a triangulation of the points thus generated. If the edge of any triangle crosses the pole curve or the self-intersection image curves, insert the intersection points; if any triangle has three pole vertices, insert its midpoint.

7. (MAP TRIANGLES) Every triangle can now have up to 2 pole vertices or base point vertices. Map each triangle onto a surface patch and clip it against the bounding box. Various types of patches result depending on the labels of a domain triangle. They are as follows:

   - All vertices are ordinary. The image of the triangle is a finite triangular patch.
   - One vertex is a pole. The image is an infinite triangular patch with one corner at infinity (Figure 9).
   - Two vertices are poles. The image is an infinite triangular patch with two corners at infinity (Figure 9).
   - One vertex is a base point. The base point blows up to a curve on the surface. Approaching the base point vertex along each of its incident edges leads to a different surface point on this curve. Thus, the image is a finite *rectangular* patch (Figure 10).
   - Combinations of ordinary, base points, and pole points. The resulting patch can be finite or infinite, with up to six sides.

   Mapping each domain triangle is accomplished by walking along its boundary and checking the vertex labels. For clipping, an iteration must be used to locate the intersection(s) of each edge of the surface patch with the bounding box. To map domain triangles with base point vertices, a parameterization of the corresponding seam curve is necessary, as explained previously and elaborated below.

Finally, we could compute surface points corresponding to complex parameter values as explained earlier. Since these points form a one-dimensional family in general (i.e. they are curves), we don't include them in the polygonal mesh; they should however be used to augment the surface display in mathematical visualization.

6.1 Discussion

We now discuss the steps in more detail.

In step 2, the curve approximation must be sufficiently linear, otherwise there is a risk that some parts of the surface that lie inside the bounding box will be missed.

In step 5, points on the unit triangle can be generated either uniformly or adaptively spaced. Points that are uniformly spaced in the s and t directions are easily generated. For instance we can generate $n(n+1)/2$ points by taking $i+1$ equally spaced points on each line $s+t = i/(n-1), i = 0, \ldots, n-1$. The points can also be selected based on local surface curvature, for instance using heuristics that test the local "flatness" of the surface using the tangents at triangle endpoints. In any event, the surface triangulation criterion forces us to generate all surface points before triangulating.

In step 4 we must find the base points by solving the equations $F_i = 0, i = 1, \ldots, 4$. This could be done by picking two of the equations, finding their common solutions, and then checking whether these are solutions of the other two equations. In [18] a method based on resultants is given for finding all base points and their multiplicities directly.

For step 6, any triangulation method [40] can be used, although some triangulations may have convenient properties.

Step 7 is complicated not only because of the many cases involved, but because we only know that the current domain triangle maps onto a single-sheeted patch. The patch can have up to 5 sides, and could twist in and out of the bounding box in a complicated way. The domain triangle should be further subdivided if necessary, using an adaptive domain sampling technique [49, 38, 32] (however, any time more domain points are added, we must re-triangulate). For instance, in [49] estimates of *Lipschitz constants* are used to decide when a portion of a surface is sufficiently linear to be approximated by a triangular facet.

Finally, base point vertices need special treatment. A domain triangle with b base point vertices maps onto a patch with $b+3$ sides. Three sides of the patch are the images of the domain triangle's three edges, and therefore tracing these sides (for clipping) is not a problem. How to trace the other b sides of the patch is not obvious, since their points can't be generated by evaluating the rational map at some domain points.

Consider a triangle with a base point vertex p. Suppose p is incident to the edges e_1 and e_2. Let the slope of the edges e_1 and e_2 be m_1 and m_2 respectively. Approaching p along the line of slope m_1 leads to one point on the surface, and approaching it along the line of slope m_2 leads to another point on the surface. Both these points lie on the seam curve corresponding to the base point. Parameterize the seam curve in terms of m, the slope of lines through the base point. Then the side of the patch that corresponds to the base point vertex can be traced by evaluating the seam curve parameterization at values between m_1 and m_2.

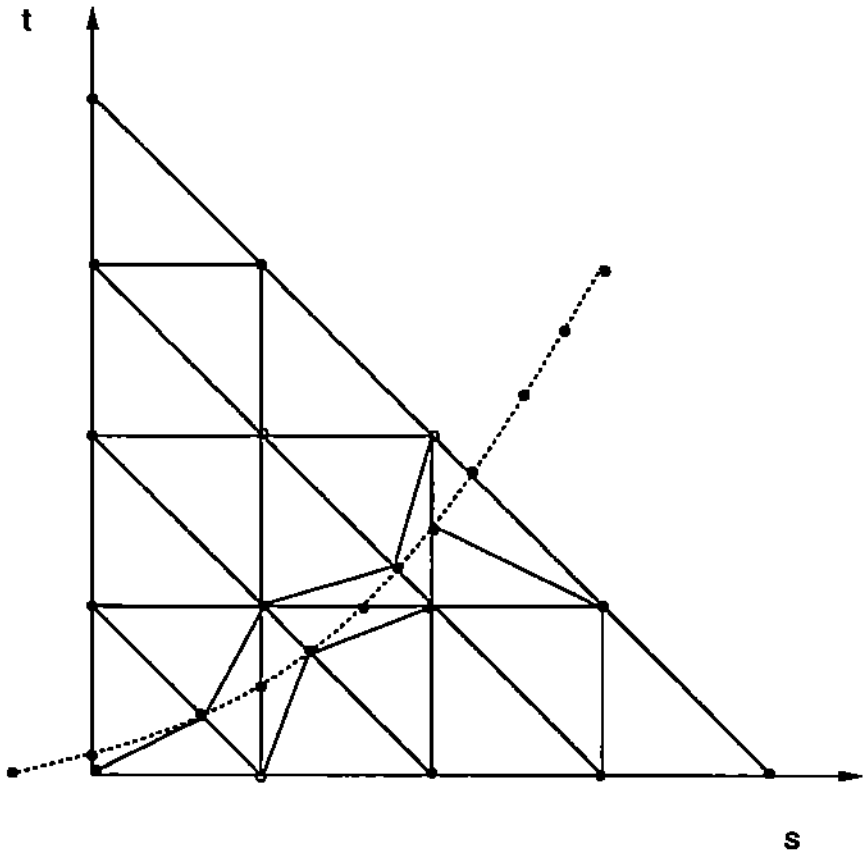

Figure 8: Partition of domain triangulation by pole curve

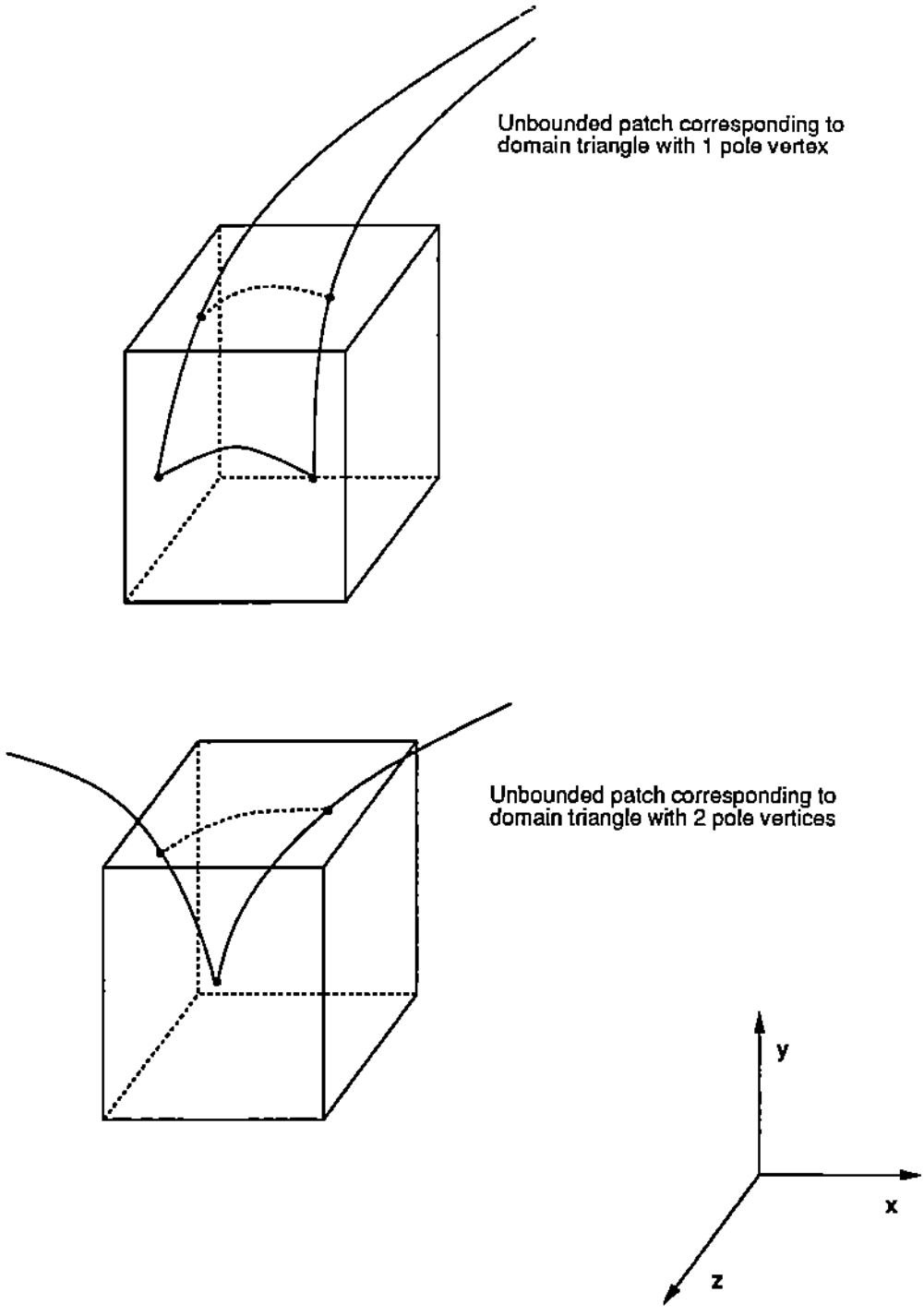

Figure 9: Image patches of domain triangles with pole vertices

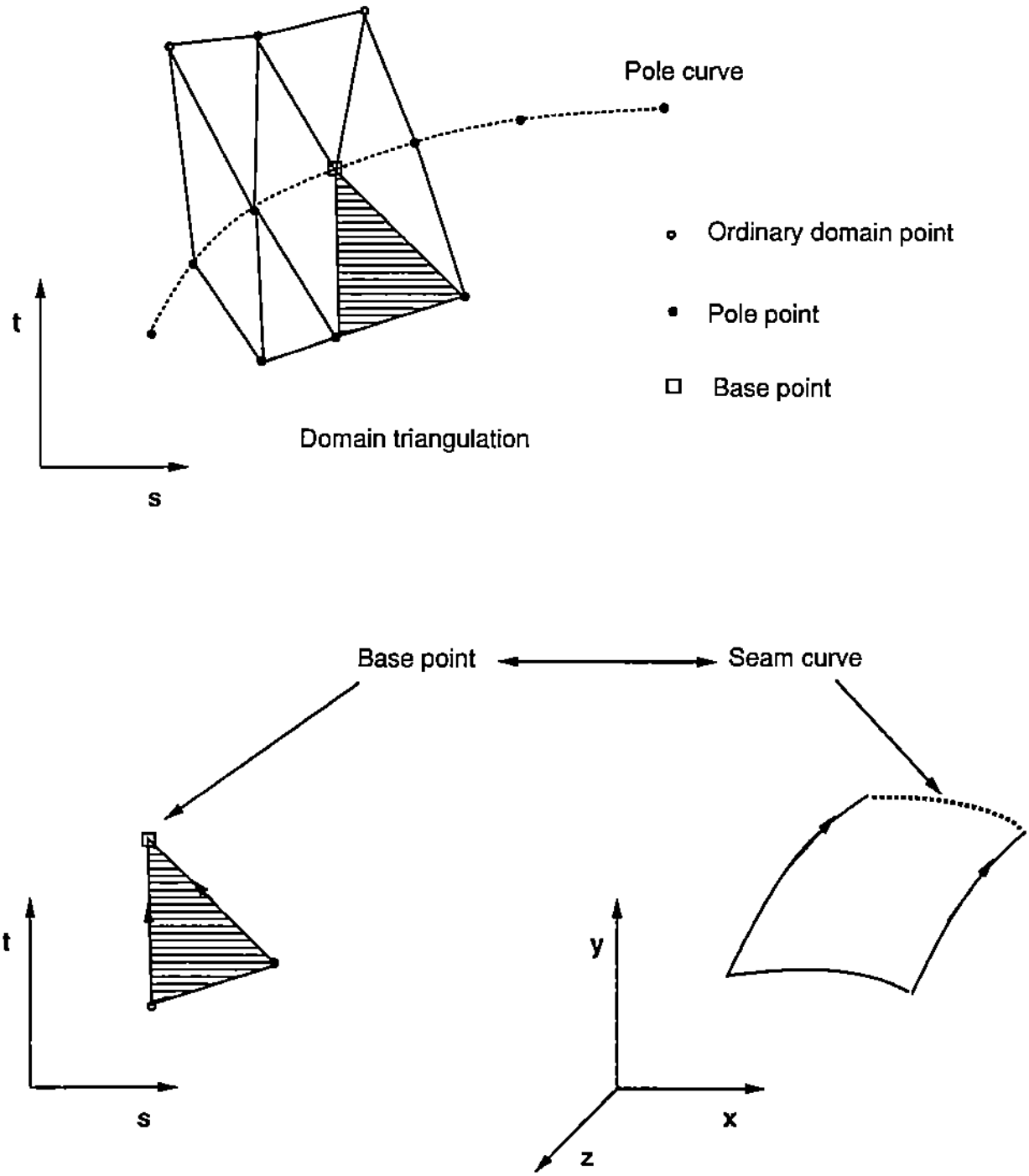

Figure 10: Image patch of a domain triangle with a base point

7 Future work

The algorithm may fail to generate an approximation that accurately represents the surface, if the pole curve subdivision is not sufficiently linear. In particular, surface features inside the bounding box may be missed. The precision to which the pole curve is approximated is at present specified by the user; more research is needed to determine how to calculate this precision automatically.

Triangulations with special properties such as the Delaunay triangulation [40] could be used to speed up step 5, which is expensive. In particular, we hope to use Delaunay properties to avoid testing edges for intersection with the pole curve, that are far away from the pole curve.

In step 7, the seam curve parameterization algorithm is necessary; however, it fails in the special case when the domain curves defined by the numerators and denominator of the rational map have identical tangents at a base point. One possible approach to solving the problem would be to use a quadratic domain transformation to derive a new parameterization of the surface where the domain curves have separate tangents at the base points.

Further work is needed in formulating numerical techniques to efficiently compute piecewise-linear approximations to the domain images of surface self-intersections.

We have implemented a significant part of the algorithm and tried various techniques for solving these problems. For instance, we also investigated a surface-trimming approach to generating topologically correct approximations to multi-sheeted surfaces. However, based on our experiments we feel that the domain partitioning approach is most suitable.

Our implementation in the Ganith system [8] was used to generate Figures 3,4, 6,7, 11,12, 13, and 14. The last four show various self-intersecting and multi-sheeted parametric surfaces.

8 Acknowledgements

This research was supported in part by NSF grants CCR 90-02228, DMS 91-01424 and AFOSR contract 91-0276.

References

[1] Arnon, D.S., (1983), "Topologically Reliable Display of Algebraic Curves," *Computer Graphics*, 17, 218-228.

[2] Arnon, D.S., (1988), "On the Display of Cell Decompositions of Algebraic Surfaces," XEROX Technical Report, 1988.

[3] Abhyankar, S. S., and Bajaj, C., (1988), Automatic Parameterization of Rational Curves and Surfaces III: Algebraic Plane Curves, *Computer Aided Geometric Design*, 5, 309 - 321.

[4] Bajaj, C., Canny, J., Garrity, T. and Warren, J., (1989), "Factoring Rational Polynomials over the Complexes," *Proceedings of the ACM SIGSAM Symposium on Symbolic and Algebraic Computation*, Portland, OR, July 17-19, 81-90, ACM Press, NY.

[5] Bajaj, C., Garrity, T. and Warren, J., (1988), "On the Applications of Multi-Equational Resultants," Technical Report CSD-TR-826, Department of Computer Sciences, Purdue University.

[6] Bajaj, C., Hoffmann, C.M., Lynch, R.E., and Hopcroft, J.E.H., (1988), "Tracing Surface Intersections," *Computer Aided Geometric Design*, 5, 285-307.

[7] Bajaj, C., and Ihm, I., (1992), "Algebraic Surface Design with Hermite Interpolation," *ACM Transactions on Graphics*, 11, 1, 61-91, January.

[8] Bajaj, C., and Royappa, A., (1990) "The Ganith Algebraic Geometry Toolkit", in *DISCO '90*, Capri, Italy. Also Computer Science Technical Report No. 91-065, Purdue University.

[9] Bajaj, C., and Royappa, A., (1992), "Robust Display of Rational Parametric Curves and Surfaces." In Proceedings of *Curves and Surfaces in Computer Vision and Graphics III*, pp 70-80, Boston, November 1992.

[10] Bajaj, C., and Royappa, A., (1993), "Finite Representations of Real Parametric Curves and Surfaces," to appear in proceedings of *Geometric Modeling in Computer Graphics*, Genova, Italy, June.

[11] Bajaj, C. and G. Xu, (1992), Piecewise Rational Approximations of Real Algebraic Curves, Technical Report CSD-TR-92-040, Department of Computer Science, Purdue University.

[12] Bajaj, C., and Xu, G., (1992), "Piecewise Approximations of Algebraic Surfaces," manuscript, Department of Computer Science, Purdue University.

[13] Bloomenthal, J., (1988) "Polygonization of Implicit Surfaces.", *Computer Aided Geometric Design*, 5, 341 - 355.

[14] Boehm, W., Farin, G., and Kahmann, J., (1984), A Survey of Curve and Surface Methods in CAGD, *Computer Aided Geometric Design*, 1, 1-60.

[15] Buchberger, B., (1985) "Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory," *Multidimensional Systems Theory*, Chapter 6, N. Bose (eds). Reidel Publishing Co.

[16] Catmull, E., (1975) "Computer Display of Curved Surfaces", in *IEEE Conference Proceedings on Computer Graphics, Pattern Recognition and Data Structures*, May 1975, reprinted in *Tutorial and Selected Readings in Interactive Computer Graphics*, H. Freeman (ed.), IEEE, 1980, 309-315.

[17] Char, B. W., Geddes, K. O., Gonnet, G. H., Leong, B. L., Monagan, M. B., and Watt, S. M., (1991), *First Leaves: A Tutorial Introduction to Maple V*, Springer-Verlag.

[18] Chionh, E., (1990), "Base Points, Resultants, and the Implicit Representation of Parametric Surfaces." Ph.D. Thesis, University of Waterloo.

[19] Chuang, J-H, and Hoffmann, C.M., (1989), "On Local Implicit Approximation and its Applications", in *ACM Transactions on Graphics*, 8, 4, 298-324.

[20] DeRose, T., (1991), "Rational Bezier Curves and Surfaces on Projective Domains", in *NURBS for Curve and Surface Design*, G. Farin, Editor, SIAM Press, 1-14.

[21] Edelsbrunner, H., (1987), *Algorithms in Combinatorial Geometry*, Springer-Verlag.

[22] Farin, G., (1991) *NURBS for Curve and Surface Design*, (Editor), SIAM Press.

[23] Fateman, R., (1992), "Honest Plotting, Global Extrema, and Interval Arithmetic," In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, Berkeley, California, July 27-29, 216-223.

[24] Field, D.A., (1992), "Delaunay Criteria for Triangulating Surfaces." In Proceedings of *Curves and Surfaces in Computer Vision and Graphics III*, pp 237-246, Boston, November 1992.

[25] Foley, J., Van Dam, A., Feiner, S.K., and Hughes, J.F., (1990), *Fundamentals of Interactive Computer Graphics*, 523-529, Addison Wesley Publishing Co.

[26] Gao, X.S., and Chou, S.C., (1990) "Implicitization of Rational Parametric Equations," Technical Report TR-90-34, Department of Computer Science, University of Texas at Austin, 1990.

[27] Geisow, A., (1983), "Surface Interrogations", Ph.D. Dissertation, School of Computing and Accountancy, University of East Anglia.

[28] Hall, M., and Warren, J., (1988), Adaptive Tessellation of Implicitly Defined Surfaces, "Adaptive Polygonization of Implicitly Defined Surfaces," in *IEEE Computer Graphics & Applications*, 10, 6, 33-42, November.

[29] Hartshorne, R., (1977), *Algebraic Geometry*, Springer-Verlag.

[30] Kaltofen, E., (1983), "Factorization of Polynomials." in *Computer Algebra, Symbolic and Algebraic Computation*, Buchberger, B., Collins, G., and Loos, R., Editors, Springer-Verlag, 94-113.

[31] Kajiya, J.T., (1982), "Ray Tracing Parametric Patches," *Proceedings of SIGGRAPH '82*, in *Computer Graphics*, 16, 3, 245-254.

[32] Kosters, M., (1991), "Curvature Dependent parameterization of Curves and Surfaces", *Computer Aided Design*.

[33] Lane, J. and Carpenter, C., (1979), "A Generalized Scan Line Algorithm for the Computer Display of Parametrically Defined Surfaces", in *Computer Graphics and Image Processing*, 11, 3, 290-297.

[34] Macaulay, F.S., (1902), "Some Formulae in Elimination," in *Proc. London Math. Soc.*, 35, 3-27, 1902.

[35] Macaulay, F.S., (1916), *The Algebraic Theory of Modular Systems*, Cambridge University Press, 1916.

[36] Manocha, D., (1992), *Algebraic and Numeric Techniques in Modeling and Robotics*, Ph.D. Thesis, University of California at Berkeley, 1992.

[37] Manocha, D., and Canny, J. F., (1991), "Efficient Techniques for Multipolynomial Resultant Algorithms," in *Proceedings of International Symposium on Symbolic and Algebraic Computation*, 271-276, 1991.

[38] Patterson, R.R., and Bajaj, C., (1989) Curvature Adjusted Parameterizations of Curves, Computer Science Technical Report CSD-TR-907 and CAPO Report CER-89-18.

[39] Petersen, C.S., (1984), "Adaptive Contouring of Three-Dimensional Surfaces," in *Computer Aided Geometric Design*, 1, 1984, 61-74.

[40] Preparata, F.,P., and Shamos, M.I., (1985), *Computational Geometry*, Springer-Verlag, 1985.

[41] Rockwood, A., Heaton, K., and Davis, T., (1989), "Real-Time Rending of Trimmed Surfaces", in *Proceedings of ACM SIGGRAPH '89*, Boston, 107-116.

[42] Schweitzer, D. and Cobb, E.S., (1982), "Scanline Rendering of Parametric Surfaces", *Computer Graphics*, 16, 3, 265-271.

[43] Sederberg, T.W. (1990), "Techniques for Cubic Algebraic Surfaces: Part One," *IEEE Computer Graphics & Applications*, pp 14-27, July.

[44] Semple, J.G., and Roth, L. (1985), *Introduction to Algebraic Geometry*, Oxford University Press, 1985.

[45] Shantz, M. and Chang, S., (1988), "Rendering Trimmed NURBS with Adaptive Forward Differencing," in *Proceedings of ACM SIGGRAPH '88*, Atlanta, 189-198.

[46] Sugihara, K., and Iri, M., (1989), "Construction of the Voronoi Diagram for One Million Generators in Single Precision Arithmetic", First Canadian Conference on Computational Geometry, Montreal, Canada.

[47] Sutherland, I.E., and Hodgman, G.W., (1974), "Reentrant Polygon Clipping", *Communications of the ACM*, 17, 1, 32-42.

[48] Tindle, G.L., (1897), "Tetrahedral Triangulation," *The Mathematics of Surfaces II*, 387-394, Ed. R.R. Martin, Clarendon Press, Oxford.

[49] Von Herzen, B., (1989), *Applications of Surface Networks to Sampling Problems in Computer Graphics*, Ph.D. Thesis, California Institute of Technology.

[50] Warren, J., (1992), "Creating Multisided Rational Bezier Surfaces Using Base Points", in *ACM Transactions on Graphics*, 11, 2, 127-139.

[51] Weiler, K. and Atherton, P., (1977), "Hidden Surface Removal Using Polygon Area Sorting", SIGGRAPH '77 Proceedings, published as *Computer Graphics* 11, 2, 214-222.

[52] Wolfram, S., (1991), *Mathematica: A System for Doing Mathematics by Computer*, Addison-Wesley.

[53] Zariski, O., (1971), *Algebraic Surfaces*, Springer-Verlag.

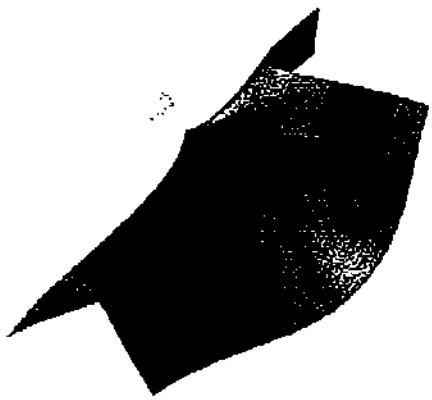

Figure 11: Rational parametric surface (cubic node)

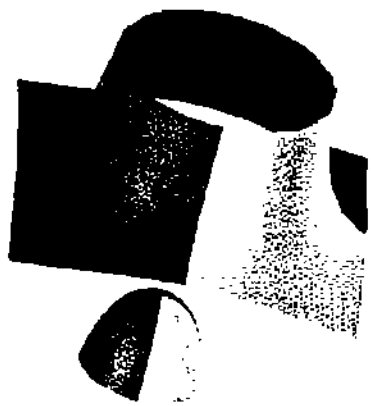

Figure 12: Rational parametric surface (two sheets)

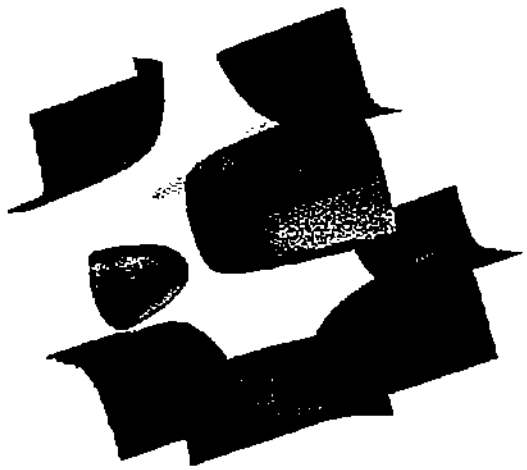

Figure 13: Rational parametric surface (nine sheets)

Figure 14: Rational parametric surface (two sheets, self-intersecting)